INFORMATION TECHNOLOGY WEEK 1979

LEARN HOW TO PROGRAM IN BASIC

CLASSES CONDUCTED AT

SWINBURNE COLLEGE OF TECHNOLOGY

INSTRUCTORS:

KATE BEHAN
RALPH TRELOAR
NOEL KELLY
ROY FRANCIS
DIANA HOLMES
NIGEL ANDERSON

The objective of these 2 classes is to show you how to write a simple computer program in BASIC.

By the end of the 2 classes you will have written several small programs. Hopefully, in doing so, you will realize that a computer is a useful tool for performing routine, repetitive tasks.

The computer you will be using is a small computer designed for teaching purposes. The version of the BASIC language is MONECS BASIC.

As it is designed to be used in the student environment you will be using mark-sense cards and printed output. Many computers have different input facilities - e.g. a keyboard with a visual display - which enables you to actually sit at the machine and enter your program directly via the keyboard.

Programming Concepts

A computer must be given a set of instructions before it can do anything.

A PROGRAM is the name we give to a set of instructions that perform any given operation.

The most important part of any program is the output that it produces.

In order to produce output, the machine must be given an instruction to do so.

If I want a computer to print the word HELLO then I must give it an instruction that says

```
PRINT "HELLO"
```

If I want the computer to say

```
HELLO KATE
```

then I would give it an instruction

```
PRINT "HELLO KATE"
```

If my name was FRED I would not be very impressed by a machine that printed

```
HELLO KATE
```

for me.

It would be a good idea to have a program that said

HELLO KATE to Kate

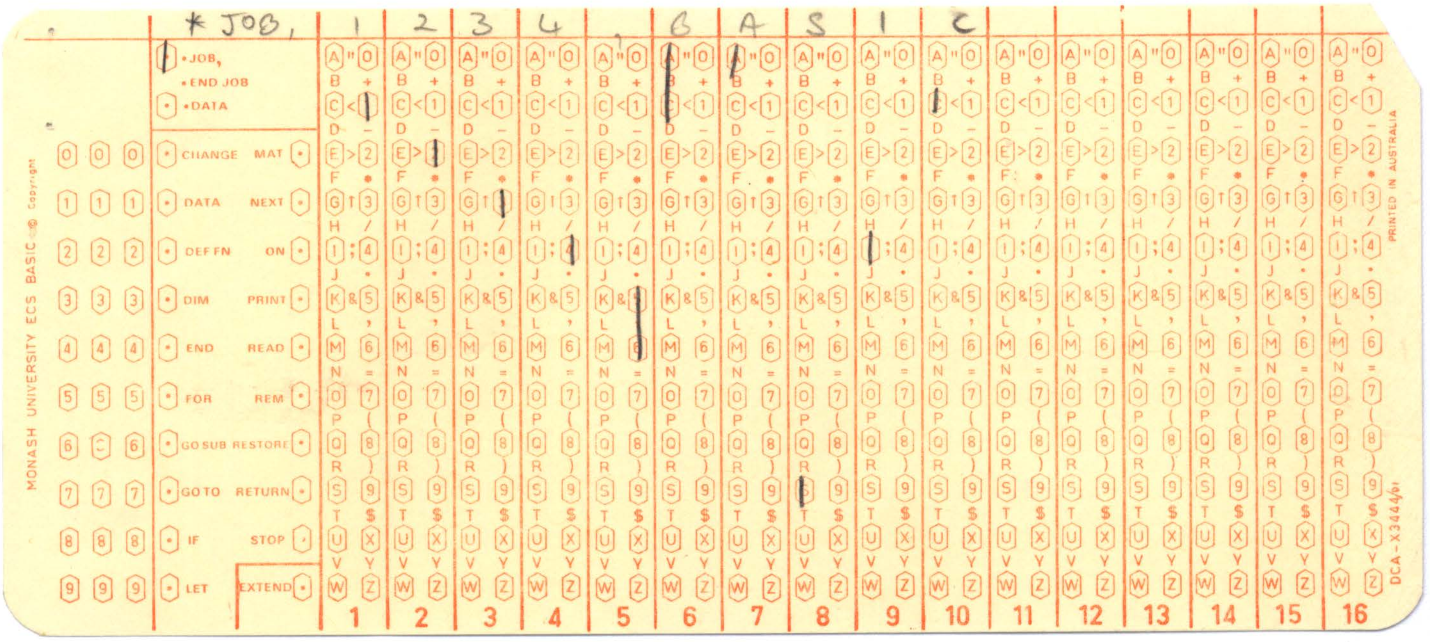
or

HELLO RALPH to Ralph

or

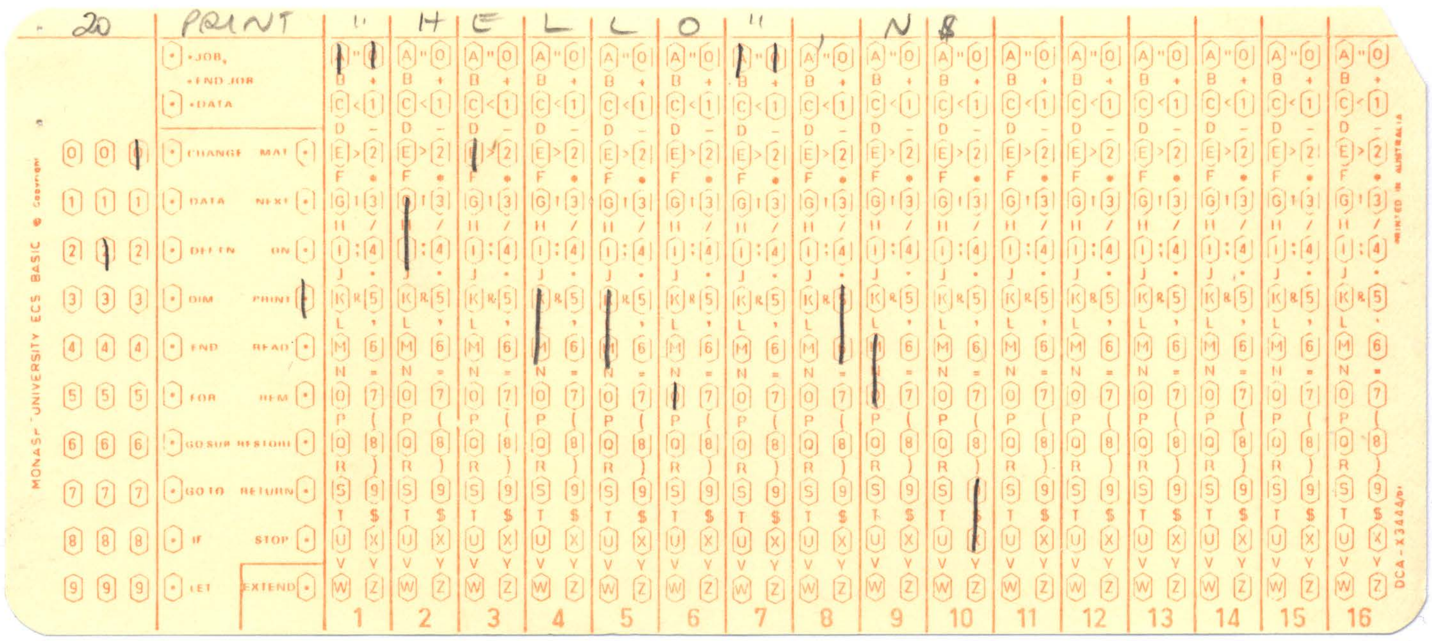
HELLO NOEL to Noel.

So, we will write a program that prints HELLO with your own name.



```
*JOB, 1234, BASIC
```

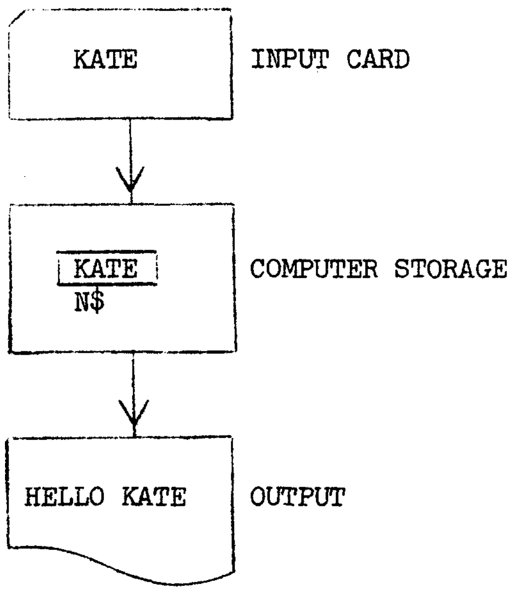
MONASH UNIVERSITY ECS BASIC © Copyright
•JOB,
•END JOB
•DATA
CHANGE MAT
DATA NEXT
DEF FN ON
DIM PRINT
END READ
FOR REM
GO SUB RESTORE
GO TO RETURN
IF STOP
LET EXTEND
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
PRINTED IN AUSTRALIA
DCA-X3444/01



```
20 PRINT "HELLO", N$
```

MONASH UNIVERSITY ECS BASIC © Copyright
•JOB,
•END JOB
•DATA
CHANGE MAT
DATA NEXT
DEF FN ON
DIM PRINT
END READ
FOR REM
GO SUB RESTORE
GO TO RETURN
IF STOP
LET EXTEND
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
PRINTED IN AUSTRALIA
DCA-X3444/01

A machine doesn't know what your name is unless you tell it - so we will need to tell it your name - that is we will give the machine some input. The input will be a card which has your name on it.



Program

```
*JOB,1234,BASIC
10  READ   N$
20  PRINT  "HELLO", N$
30  DATA   "KATE"
40  END
*END JOB
```

Things to note:

1. Each instruction is on a separate card.

2. Each instruction has a line number - these must be in ascending order.

3. You need 2 special cards to make your program work on our computer:-

*JOB,1234,BASIC	is the first card which tells the computer that the instructions to follow are in the BASIC language.
*END JOB	is the last card which tells the computer that it has now finished with your program.

4. Cards must be marked carefully or the machine cannot read them.

5. In the program you are about to write you have used one input variable - N$ and one constant - "HELLO".

   Constants must be enclosed in inverted commas.

   Variables can be numeric or not numeric. Our variable - N$ - is not numeric and the $ after the N tells the machine this. Numeric variables do not have a $ after the symbol.

   e.g. variables called

   A or B or C or D

   would be numeric variables whilst

   A$ or B$ or C$ or D$

   would be non-numeric variables.

EXERCISE 1

Mark the 6 cards necessary for the program to print HELLO with your name.

Get an instructor to check your cards and show you how to use the computer in BA406 to run your program.

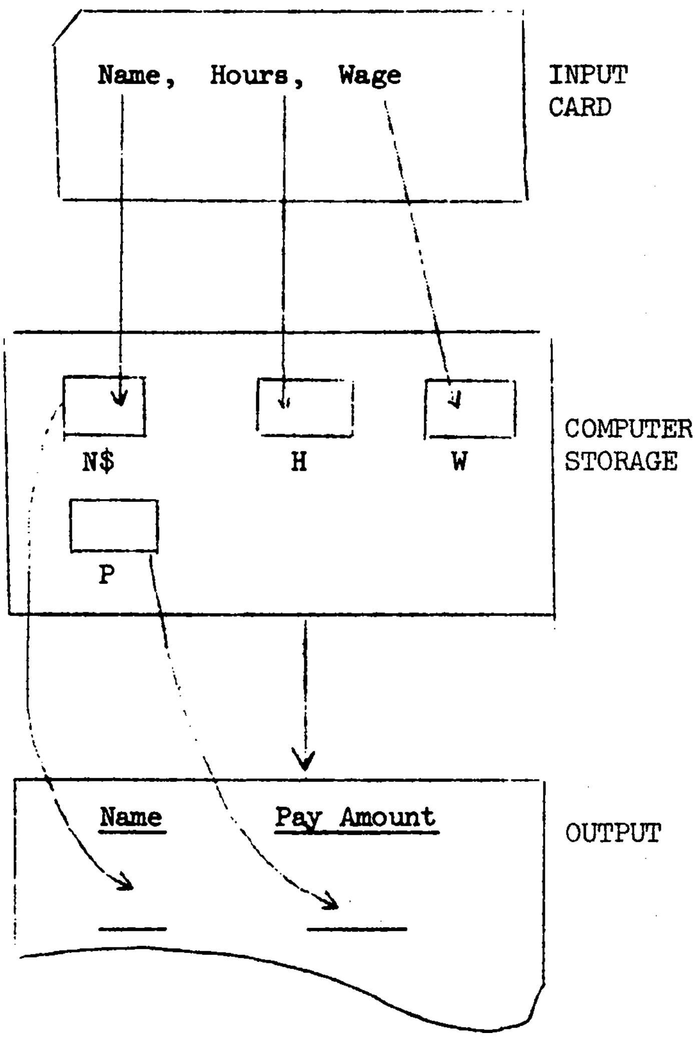
Do not be despondent if the program does not work at the first run as programs hardly ever do.

If you have made an error, the machine will indicate where the error is. You should fix it and then re-run your program.

PROGRAM ILLUSTRATION 1

Here is a program that calculates weekly pay for employees. The output we want is shown below.

Our input card will contain name, hours worked and wage rate.



The Program

```
*JOB,1234,BASIC
10  PRINT "NAME", "PAY AMOUNT"
20  READ N$, H, W
30  LET P = H * W
40  PRINT N$, P
50  DATA "KATE", 40, 100
60  END
*END JOB
```

Things to note:

Statement 20 reads the value of name, hours worked and wage rate into the computer.

Statement 30 says: We need a new storage area called P (Pay) and let it be equal to hours worked X wage rate. We use * symbol for multiplication.

Statement 10 prints a heading line so we know that is being printed out.

EXERCISE 2

Write a program, based on the above to pay yourself whatever you think you deserve.
The biggest amount the computer can print is 7 digits.

PROGRAM ILLUSTRATION 2

In program illustration 1 we wrote a program that paid 1 person. We will now modify that program to pay any number of people.

```
*JOB,1234,BASIC
05  PRINT "NAME", "PAY AMOUNT"
10  READ N$, H, W
15  IF N$ = "XX" THEN 150
20  LET P = H * W
40  PRINT N$, P
45  GO TO 10
50  DATA - (add your own data here)
70  DATA
80
90
100
110
120
130
140 DATA "XX", 0, 0
150 END
*END JOB
```

Things to note:

Statement 5 -	we print the heading first before we read any data.
Statement 15 -	we must have a way of getting to the end of the program so we put in a dummy record to indicate the end of the data.
Statement 45 -	when we reach this statement we return to statement 10. Each time we return to statement 10 a new data card will be read and new values for N$, H and W become available.
Statement 140 -	is the dummy record which indicates the end of the data.

EXERCISE 3

Modify your exercise 2 so that you can now pay yourself plus 7 other friends, relatives, employers etc., whatever you feel they deserve.

PROGRAM ILLUSTRATION 3

We want to produce a listing of salesman's commissions.

The salesman receives 10% of total sales as commission plus a bonus calculated as follows:

Sales	Bonus
Over 2,000	5% of total sales
Over 1,000	2% of total sales
1,000 or less	no bonus

The output is to show, for each salesman,

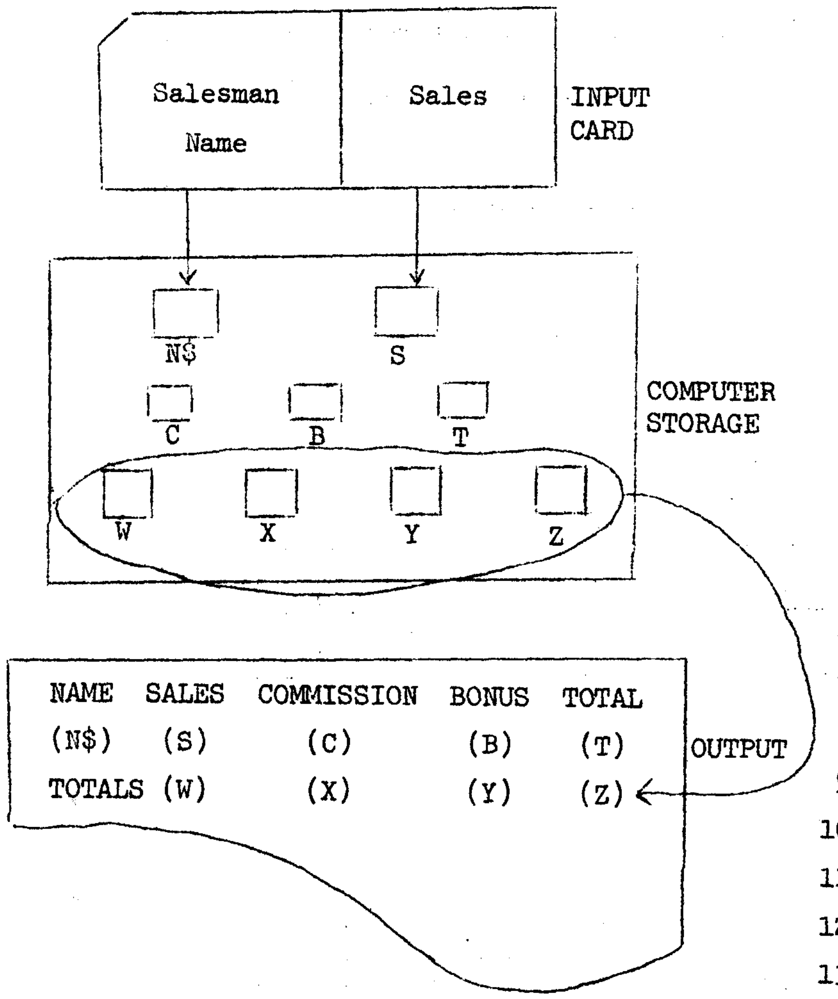
Salesman name, Total Sales, Commission, Bonus, Total Due.

Also we want a total line which shows

All Sales, All Commissions, All Bonuses, All Total Dues.

The input card contains

Salesman name and total sales for that Salesman.



Program

```
*JOB,1234,BASIC
 04  PRINT "NAME","SALES","COMMISSION",
           "BONUS","TOTAL"
 06  LET W = 0
 07  LET X = 0
 08  LET Y = 0
 09  LET Z = 0
 10  READ N$, S
 15  IF N$ = "XX" THEN 160
 20  IF S > 2000 THEN 50
 30  IF S > 1000 THEN 70
 40  LET B = 0
 45  GO TO 80
 50  LET B = .05 *S
 60  GO TO 80
 70  LET B = .02 *S
 80  LET C = .1 *S
 90  LET T = C + B
100  PRINT N$, S, C, B, T
110  LET W = W + S
120  LET X = X + C
130  LET Y = Y + B
140  LET Z = Z + T
150  GO TO 10
160  PRINT "TOTAL", W, X, Y, Z
170  DATA "JOE", 2500
180  DATA "FRED", 500
190  DATA "LIZ", 3000
200  DATA "NOEL", 2000
210  DATA "DIANA", 2500
220  DATA "KATE", 1500
230  DATA "RALPH", 2800
240  DATA "NIGEL", 400
250  DATA "XX", 0
260  END
 * END JOB
```

EXERCISE 4

Write a program that produces a listing of discounts offered to customers.

The amount of discount is based on the dollar amount of each purchase as follows:

Purchases	Discount Rate
> $ 500	1%
> $1000	2%
> $2000	5%

The output should be as follows:

Customer Name	Purchase Amount	Discount	Amount Owing
FRED	> $500	1%	$5
TOM	> $2000	5%	$100.
Totals	$2500	6%	$105.

The input card contains:

Customer name and Purchase amount.

(Note: the discount is based on each purchase - not on accumulated purchases. If Fred has 2 purchases of 520 each then he receives 1% on each purchase - the idea of the discount is to encourage fewer sales of larger amounts thus reducing overheads).

EXERCISE 5

Write a program that reads payroll data and produces a report showing the total of pay for each department. The cards contain:

. Employee number.

. Department number.

. Pay.